

VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

Zadání bakalářské práce

Student:

David Katanik

Studijní program:

B2647 Informační a komunikační technologie

Studijní obor:

2612R025 Informatika a výpočetní technika

Téma:

Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování:

čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: Tieto Czech s.r.o.
2. Struktura závěrečné zprávy:
 - a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
 - b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
 - c) Zvolený postup řešení zadaných úkolů.
 - d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
 - e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
 - f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Dohnálek**

Konzultant bakalářské práce: Ing. Radoslav Glajc

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016



doc. Dr. Ing. Eduard Sojka
vedoucí katedry



prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 14. dubna 2016


.....

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 14. dubna 2016

.....

Tieto Czech s.r.o.
28. října 3346/91
702 00 Ostrava - Moravská Ostrava
IČO 64608051 DIČ CZ64608051

Rád bych poděkoval společnosti Tieto Czech s.r.o. za umožnění absolvování individuální odborné praxe, zvláště pak celému týmu lidí, kteří pracují na produktu Tieto TRIP.

Mé poděkování patří také Ing. Pavlu Dohnálkovi za příkladné vedení práce, věcné připomínky a spolupráci.

Abstrakt

Cílem této bakalářské práce je popsat průběh mnou vykonané individuální odborné praxe ve společnosti Tieto Czech s.r.o. na pozici Software Developer. Obsahem mé práce ve společnosti byl vývoj webových aplikací výhradně na platformě Java.

V této práci popisuji společnost spolu s její historií. Následně se zaměřuji na jednotlivé nekomerční projekty, které slouží pro uvedení nově příchozích zaměstnanců do společnosti. Dále se věnuji komerčnímu projektu Tieto TRIP. Tento jedinečný produkt, který je starší více jak dvě desítky let, kombinuje NoSQL databázi s vyhledávacím strojem. V této kapitole popisuji práci, kterou jsem vykonal, a můj přínos produktu, mezi který patří zrychlení implementace, a tím ušetření nákladů. V závěrečné části shrnuji výhody absolvování odborné praxe a použité znalosti získané výukou.

Klíčová slova: Individuální odborná praxe, Tieto, Tieto TRIP, NoSQL databáze, Vyhledávací stroj, Webové aplikace, Java, Spring framework

Abstract

The aim of this bachelor thesis is to describe process of my individual professional practice in the Tieto Czech s.r.o. company on position of a Software Developer. The content of my work in the company was development of web applications exclusively on Java platform.

In this work I describe the company, along with its history. Then I focus on particular non-commercial projects, which serve as an introduction for the new employees into the company. Furthermore I focus on the commercial project called The Tieto TRIP. This unique product which is older more then two decades combines NoSQL database with search engine. In this chapter I describe work that I have done and my contribution to the product which includes acceleration of implementation and thus save costs. In the final part I summarize the advantages of completion of professional practice and usage of knowledge gained by education.

Key Words: Individual Professional Practice, Tieto, Tieto TRIP, NoSQL database, Search engine, Web applications, Java, Spring framework

Obsah

Seznam použitých zkratek a symbolů	8
Seznam obrázků	9
Seznam výpisů zdrojového kódu	10
1 Úvod	11
2 Tieto	12
2.1 Tieto Czech s.r.o.	12
2.2 Pracovní zařazení	12
3 Nekomerční projekty	13
3.1 Hardware Index	13
3.2 Quizzer	14
3.3 Tieto Component shop	17
4 Komerční projekt	19
4.1 Tieto TRIP	19
5 Závěr	28
5.1 Uplatněné a scházející dovednosti	28
5.2 Zhodnocení odborné praxe a jejích výsledků	28
Literatura	29

Seznam použitých zkratek a symbolů

API	– Application Programming Interface
CSS	– Cascading Style Sheets
GUI	– Graphical User Interface
HTML	– HyperText Markup Language
IDE	– Integrated Development Environment
JPA	– Java Persistence API
JSF	– JavaServer Faces
JSP	– JavaServer Pages
JSON	– JavaScript Object Notation
LDAP	– Lightweight Directory Access Protocol
MMC	– Microsoft Management Console
MVC	– Model View Controller
NoSQL	– Non Structured Query Language
OEM	– Original Equipment Manufacturer
POM	– Project Object Model
SQL	– Structured Query Language
UML	– Unified Modeling Language
XML	– Extensible Markup Language
XSLT	– eXtensible Stylesheet Language Transformations

Seznam obrázků

1	Hardware Index - Hardware detail	14
2	Quizzer - Přiřazené testy	15
3	Quizzer - Přiřazování testů	15
4	Quizzer - Průběh testu	16
5	Quizzer - Kontrola výsledků testu	16
6	Tieto Component shop - Hlavní stránka	17
7	Tieto Component shop - Vytváření nového příspěvku	18
8	Konfigurační framework - Formulář pro pokročilé vyhledávání	22
9	Tieto TRIP GUI - Základní vyhledávání slova	26
10	Tieto TRIP GUI - Detail vyhledané položky slova	27
11	Tieto TRIP GUI - Expertní mód	27

Seznam výpisů zdrojového kódu

1	Konfigurační framework - Formulář pro pokročilé vyhledávání	23
2	Konfigurační framework - Implicitní nastavení pro GUI	24

1 Úvod

Hlavním argumentem pro výběr individuální odborné praxe namísto klasické bakalářské práce, bylo propojení nabytých teoretických znalostí s praktickými, a to jak v poli návrhu informačních systémů, tak i různých agilních metodik vývoje. Další nesmírnou výhodou odborné praxe je seznámení se s mnoha pracovními postupy a kulturou ve společnosti.

Do společnosti jsem nastoupil na pozici Software Developer a náplní mé práce bylo vytváření různých webových aplikací pro interní potřeby firmy. Veškeré tyto aplikace byly implementovány na platformě Java s využitím otevřených softwarů. Během vývoje těchto aplikací jsem se měl seznámit s firemními postupy, vyzkoušet si většinu používaných technologií a naučit se pracovat v týmu. Po určitém čase jsem byl přesunut do týmu pracujícím na projektu Tieto TRIP. Tento unikátní produkt spojuje NoSQL databázi s vyhledávacím strojem. V tomto týmu jsem pracoval na tvorbě grafického uživatelského rozhraní, sloužícího jako nadstavba dosavadnímu projektu. Zapojil jsem se do implementace i aplikací nepřímo souvisejících s grafickým rozhraním.

Tato práce tedy popisuje projekty, na kterých jsem pracoval, dokumentuje mnou vykonanou práci, popisuje různé technologie a pracovní postupy, se kterými jsem přišel do styku. Práce shrnuje mé nabyté zkušenosti prostřednictvím individuální odborné praxe a poukazuje na mé znalosti, které jsem během studia dosud nezískal.

2 Tieto

Společnost vznikla v roce 1968 ve finském Espoo pod tehdejšími názvem Tietoehdas Oy, kdy ze začátku obstarávala vývoj a správu IT systémů pro finskou Union Bank a její klienty a pro lesní průmysl. Společnost se následně rozšiřovala a začala se zaměřovat taktéž na osobní počítače a vývoj IT systémů.

Rozkvětu společnosti v 90. letech pomohly zejména akvizice, fúze a vstup do strategických aliancí, kdy se během několika let měnil mnohokrát název, a to v roce 1995 na TT Tieto, dále v roce 1998 na Tieto a od roku 1999 pod jménem TietoEnator. Od 26. března 2009 nese společnost název **Tieto Corporation**.

V minulém desetiletí začala společnost s expanzí na mezinárodní půdě, kdy v roce 2004 otevřela první pobočku mimo Skandinávii, a to v České republice [1].

2.1 Tieto Czech s.r.o.

Společnost vstoupila do České republiky v roce 2001 a následně v roce 2004 otevřela softwarové centrum v Ostravě, kde zaměstnává více než tisíc devět set zaměstnanců. S takovýmto počtem zaměstnanců je Ostravská pobočka třetí největší pobočkou Tieto korporace na světě [1].

2.2 Pracovní zařazení

Pro absolvování individuální odborné praxe ve společnosti Tieto Czech s.r.o. jsem musel projít různými přijímacími postupy a nutnými školeními.

První kolo přijímacího pohovoru jsem absolvoval již 4. května 2015, kdy jsem se setkal s personální pracovnící. Na tomto setkání jsem prokazoval znalosti z mého životopisu a diskutoval o různých osobních záležitostech. Následně jsem psal písemný test z programovacího jazyku Java.

Ve druhém kole přijímacího řízení, konaného 14. května 2015, jsem se sešel s manažerem týmu, který prozkoušel mou technickou odbornost, a obdržel jsem první informace o mé budoucí pozici.

Dne 1. 8. a 2. 8. 2015 jsem absolvoval vstupní školení s informacemi o společnosti, týmech, lidech a celkově o náplni práce, kterou jsem měl vykonávat. Následně jsem na denní bázi začal pracovat na nekomerčních projektech.

Během celé mé praxe jsem spolupracoval spolu s dalším studentem Ludvíkem Hobzou. Na veškerých projektech, na kterých jsem s uvedeným studentem spolupracoval, jsem se aktivně podílel formou implementace, revize kódů, navrhování řešení, párového programování apod.

3 Nekomerční projekty

Nekomerčními projekty se myslí projekty, na kterých jsem se měl naučit určitou technologii a propojit získané vědomosti z univerzity. Neméně důležitou součástí vývoje těchto aplikací byla práce v týmu, sžití se s firemní kulturou a pracovními postupy v ní. Veškeré tyto zkušenosti jsem musel získat právě z interních prací pro společnost, které byly vedeny zkušenými a proškolenými zaměstnanci.

3.1 Hardware Index

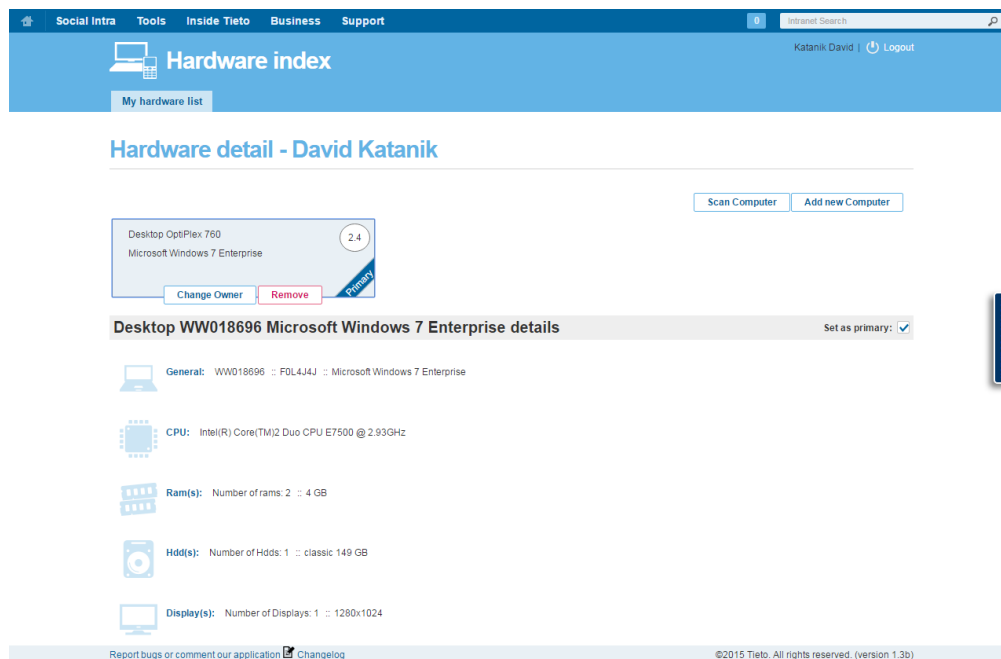
Prvním mým projektem, po nástupu do společnosti, se stala webová aplikace *Hardware Index*. Tato aplikace shromažďuje informace o hardwaru (operační paměť, paměti, procesor atd.) jednotlivých zaměstnanců a dokáže ohodnotit kvalitu jednotlivých komponent, díky čemuž je možné zobrazovat různé žebříčky a statistiky. Tato funkcionalita je důležitá zejména pro vedoucí manažery, kdy manažer potřebuje sledovat kvalitu prostředků svých zaměstnanců, aby popřípadě mohl objednávat různé části méně výkonného hardwaru.

K tomuto projektu jsem se připojil téměř u konce vývoje, a proto mi nebylo umožněno se přímo podílet na implementaci nejdůležitější části systému, avšak byl mi přidělen úkol pro rozšíření funkčnosti v podobě komponenty. V této části jsem se poprvé setkal se Spring frameworkem, který poskytuje komplexní podporu infrastruktury pro vývoj Java aplikací [2]. Na této komponentě, jejíž pracovní název je *Importer*, jsem pracoval s dalším studentem a hlavním úkolem bylo umožnit import dat z XLS souborů do systému.

Prvním řešením zadaného úkolu bylo použití knihovny Jxls Reader, která za pomoci XML konfiguračního souboru určuje, jak má být XLS soubor převeden. XML soubor definuje pevnou strukturu vstupního XLS souboru. Jelikož jsem se měl na tomto projektu naučit co nejvíce technologií, tak se neustále upravovalo zadání úkolu. Namísto pevné struktury XLS souboru byla požadována generická struktura. Generická, ve smyslu různého pořadí jednotlivých sloupců, kde není pevně daná pozice prvku v hlavičce tabulky, například prohozené sloupce CPU a GPU. Kvůli této změně požadavků bylo nutné změnit také knihovnu, na které bylo toto řešení postavené.

Druhé řešení bylo pomocí POI-XSSF, které umožňuje vytvářet, upravovat, číst a zapisovat XLS soubory. Poskytuje jednoduchou práci s těmito soubory a zajišťuje velikou škálu použití. Během implementace tohoto řešení jsem se pravidelně účastnil refaktoringu kódu, který mě naučil lépe strukturovat kód, rychleji pracovat s IDE a setkal jsem se s několika návrhovými vzory.

Nejdůležitější použité technologie: Java 1.8, Spring framework, Jxls Reader, Apache POI, Primefaces a další



Obrázek 1: Hardware Index - Hardware detail

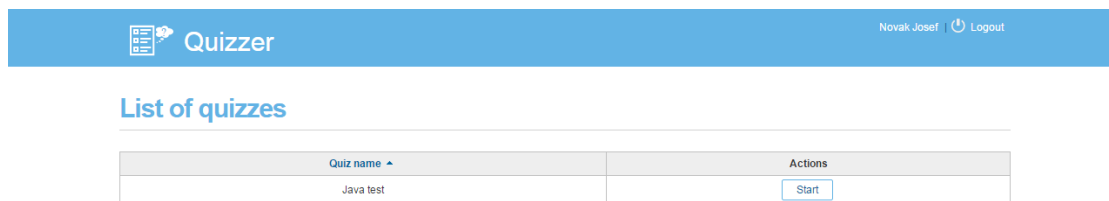
3.1.1 Shrnutí a časové vyjádření

Na tomto projektu jsem strávil přibližně dva týdny, tzn. deset pracovních dní, kdy jsem se učil zejména práci v týmu, metodiky vývoje, různé technologie a poznával podrobněji jazyk Java. Během vývoje jsem pravidelně navštěvoval SCRUM mítinky, kde jsem popisoval zvolené pracovní postupy, naučil jsem se využívat při práci Git, seznámil jsem se IDE Spring Tool Suite, který je založený na distribuci IDE Eclipse [3], a absolvoval jsem školení na jUnit testování s využitím Mockito frameworku. Ukázka hlavní stránky z pohledu uživatele na obrázku 1: Hardware Index - Hardware detail.

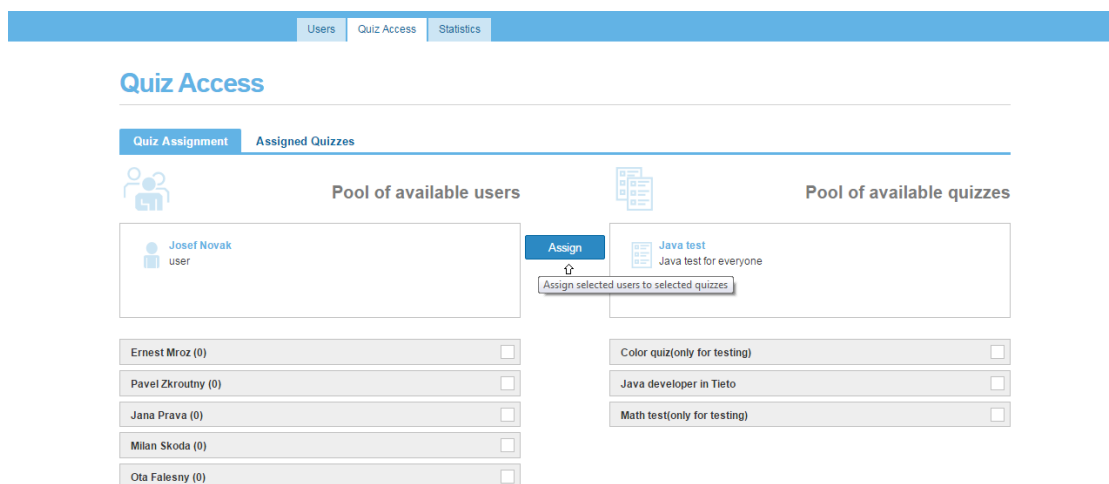
3.2 Quizzer

Druhou webovou aplikací, na které jsem se mohl podílet, byl *Quizzer*. Tato aplikace byla určena pro testování nově příchozích zaměstnanců. Každý potencionální zaměstnanec musí vykonat určitý test, a předvést tak své dovednosti. Tato aplikace měla doposud papírový test zdigitalizovat a automaticky vyhodnotit uzavřené otázky, čímž by ulehčila práci náborářům společnosti.

Každému potencionálnímu zaměstnanci je udělen identifikační kód, pod kterým se může přihlásit do systému spolu s automaticky vygenerovaným heslem. Následně přihlášený uživatel vidí testy (obrázek 2: Quizzer - Přiřazené testy), které mu byly přiřazeny prostřednictvím administrátorské stránky pro přiřazování testů (obrázek 3: Quizzer - Přiřazování testů). Každý test může obsahovat otevřené (volně psaný text) nebo uzavřené otázky (s jednou či více správnými hodnotami a možností odpočtu bodů za nesprávnou odpověď), které musí zvládnout bez možnosti vrátit se zpět k otázce, a to za určitý časový limit.



Obrázek 2: Quizzer - Přiřazené testy



Obrázek 3: Quizzer - Přiřazování testů

Na tomto projektu jsem pracoval od počáteční fáze, kdy bylo nutné specifikovat požadavky aplikace. Taktéž jsem se účastnil návrhu uživatelského rozhraní. Následně jsem se podílel na implementaci určitých funkčních bloků webové aplikace, kde má role spočívala výhradně v participaci na vývoji evaluačního algoritmu, průběhu testů (ukázka testu na obrázku 4: Quizzer - Průběh testu) a zobrazování výsledků testů, které je možné vidět na obrázku 5: Quizzer - Kontrola výsledků testu. Během implementace těchto úkolů bylo nutné psát na tyto algoritmy automatické unit testy s využitím Mockito frameworku pro mockování. Veškeré tyto úkoly jsem dělal spolu s dalším studentem za pomoci senior programátora, kdy jsme se měli společně učit správným postupům při programování a řešení úkolů celkově. Dále jsem pracoval na vytváření jednotlivých stránek a seznamoval se s technologií Primefaces, která se snadno používá, nezatěžuje aplikaci zbytečnými závislostmi a nevyžaduje žádnou konfiguraci [4]. V průběhu vývoje bylo uspořádáno několik demonstrací funkčnosti aplikace pro konkrétnější specifikace požadavků, následně byla aplikace nasazena na firemní interní server.

Pomocí párového programování jsem se naučil strukturování balíčků, správné názvosloví tříd, a absolvoval jsem několik školení vedených zkušenými programátory na programovací jazyk Java (streamy, Lambda výrazy, Method reference, psaní JavaDocu, dědičnosti, použití generických typů), Spring Roo, HTML5, CSS, Javascript, Jira a další obecné věci v programování. Tento projekt byl vytvořen pomocí Apache Maven, jakožto softwaru pro řízení projektů [5].

The screenshot shows the Quizzer application interface. At the top, there is a blue header with the Quizzer logo and the user name 'Novak Josef' with a 'Logout' link. The main content area displays a question titled 'Define immutable object?'. Below the title, there are three radio button options:

- ☐ An immutable object is not thread safe
- ☐ An immutable object can not be changed once it is created.
- ☐ An immutable object can be changed only if it is static.

At the bottom of the question area, there is a 'Next' button.

Obrázek 4: Quizzer - Průběh testu

Nejdůležitější použité technologie: Java 1.8, Spring framework ve verzi 4.1.7.RELEASE, Nástroje pro rychlý vývoj Spring Roo verze 1.3.1.RELEASE., Jako databázi jsme využili PostgreSQL ve verzi 9.1-901-1.jdbc4., Hibernate verze 4.3.11.Final, Primefaces verze 5.2, Spring security verze 4.0.2.RELEASE a další.

3.2.1 Shrnutí a časové vyjádření

Celkově jsem pracoval na tomto projektu okolo šesti týdnů, tzn. třicet pracovních dní, kdy během těchto dní jsem se seznámil s různými agilními metodikami vývoje, vyvíjel jsem průběh testování, evaluační algoritmy, zobrazování výsledků, několik webových stránek prostřednictvím Primefaces, psal unit testy za použití Mockito frameworku a poznával Spring Framework spolu se Spring Roo. Celý projekt byl spravován a řízen prostřednictvím softwaru Jira společnosti Atlassian.

The screenshot shows the Quizzer application interface displaying quiz results. At the top, there is a blue header with the Quizzer logo and the user name 'Novak Josef' with a 'Logout' link. The main content area displays a table titled 'Quizzes taken by Josef Novak [user]'. The table has two columns: 'Quiz name' and 'Quiz date'. The first row shows 'Java test' and '31.03.2016'. Below the table, there is a 'Quiz results' modal window. The modal window displays the quiz title 'Java test', the quiz result '10/30 (-17.0%)', and the quiz date '31.03.2016'. The modal window also displays a list of questions and their answers:

Options:	Score	Status
String.	-8	<input type="checkbox"/>
int.	2	<input checked="" type="checkbox"/>
long.	2	<input checked="" type="checkbox"/>
byte.	2	<input checked="" type="checkbox"/>

Below the table, there is a section for 'Evaluation' and 'Duration in seconds'.

Evaluation: 6/6
Duration in seconds: 9

Below the evaluation section, there is a list of questions and their answers:

- Give some features of Interface?
- Define immutable object?
- Why is Java Architectural Neutral?

At the bottom of the modal window, there is an 'Ok' button.

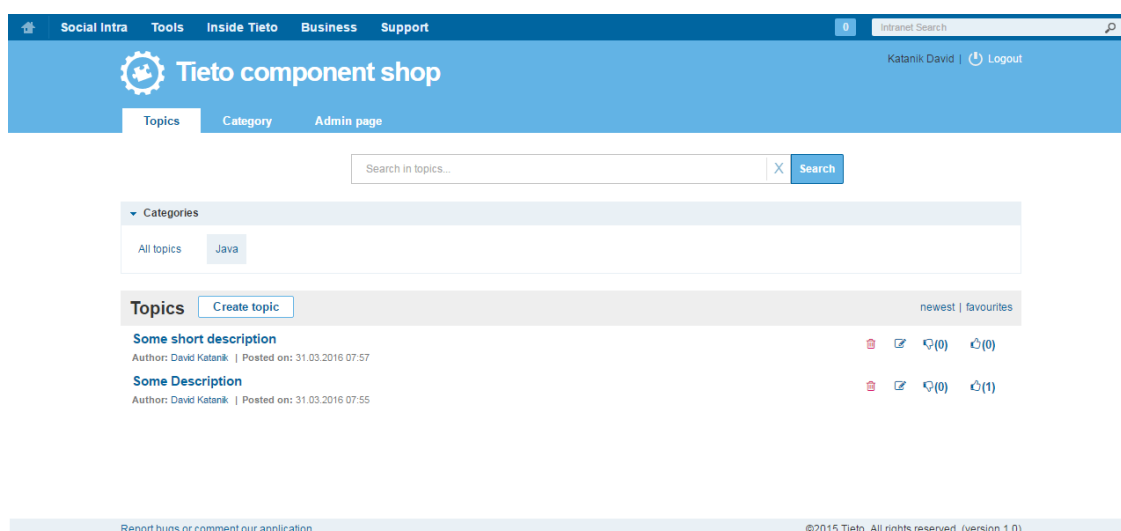
Obrázek 5: Quizzer - Kontrola výsledků testu

3.3 Tieto Component shop

Třetím nekomerčním projektem, na kterém jsem se podílel, byl *Tieto Component shop*. Z názvu se může zdát, že jde o e-shop nebo jinou podobnou aplikaci, avšak není tomu tak. Aplikace měla sloužit jako repozitář zajímavých softwarových řešení pro potřeby sdílení informací a postupů mezi zaměstnanci. Ti mohli přidávat nová témata, pomocí vyhledávače hledat požadovaná řešení, hodnotit a komentovat cizí příspěvky. Celý tento projekt jsem zpracovával spolu s dalšími dvěma studenty, kde každý měl svou roli v týmu.

Na základě požadavků a společné diskuze v týmu jsme rozhodovali, jaké zvolíme technologie, postupy a strukturu aplikace. Zvolili jsme, že aplikace bude implementována pomocí Spring Roo, protože dopomáhá k rychlému vývoji, a právě to byl klíčový požadavek zadavatele spolu s použitím Spring frameworku a Spring Security, napojením na firemní LDAP, zahrnutím open source fulltextového vyhledávače a využitím Primefaces.

Má práce zahrnovala zejména implementaci back-end části aplikace. Nejprve obnášela tvorbu tříd pomocí námi definovaného doménového modelu, kde vztahy mezi třídami byly vyjádřeny pomocí anotací z JPA. Použitím Spring Roo nebylo nutné vytvářet jakoukoliv databázovou vrstvu s SQL dotazy. Právě proto je Spring Roo snadno použitelný pro rychlé vytváření podnikových aplikací na platformě Java [6]. Následně bylo mým úkolem implementovat přihlašování uživatelů (pomocí Spring Security) a napojení na firemní LDAP, avšak pouze pro první přihlášení, jelikož nebylo potřeba si uchovávat veškerá jeho osobní data (pouze login, jméno a příjmení). Každé další přihlášení probíhalo z naší databáze. Další úkol byl v zakomponování otevřeného vyhledávacího enginu Apache Lucene, který je možno vidět na obrázku 6: Tieto Component shop - Hlavní stránka. Během těchto úkolů jsem pomáhal spolupracovníkům i ve front-end části aplikace na vzhledu, JSF stránkách a celkově funkcionalitě webové aplikace, kterou je možné ilustrovat obrázkem 7: Tieto Component shop - Vytváření nového příspěvku.



Obrázek 6: Tieto Component shop - Hlavní stránka

Během zadávání tohoto projektu nám bylo řečeno, že si máme určit člověka, který bude za tento projekt zodpovědný. Kolegové určili mě, a proto jsem musel plnit i administrativní práci na projektu. Mým úkolem byla koordinace práce, která zahrnuje definování dílčích úkolů, přiřazování těchto úkolů programátorům, zapisování odpracovaných hodin, komunikaci se zadavatelem a další. Abych mohl tuto práci vykonávat, musel jsem se naučit pracovat se softwarem pro řízení projektů Atlassian JIRA [7].

Nejdůležitější použité technologie: Java 1.8, Spring framework ve verzi 4.2.1.RELEASE, Spring Roo verze 1.3.1.RELEASE., Apache Lucene ve verzi 5.3.1, PostgreSQL ve verzi 9.1-901-1.jdbc4, Hibernate verze 5.0.2.Final, Primefaces verze 5.2, Spring security verze 4.0.2.RELEASE a další

3.3.1 Shrnutí a časové vyjádření

Tento projekt jsem vypracovával s dalšími dvěma studenty v průběhu září a října roku 2015, kdy jsme prošli společně celým cyklem vývoje aplikace. Společně jsme strávili na tomto projektu přibližně sedm pracovních dní, avšak byl to první projekt, na kterém jsme mohli pracovat i ve volném čase. Projekt obsahoval několik zajímavých úkolů od fulltextového vyhledávače Apache Lucene, až po práci se Spring Roo. Vyzkoušel jsem si také roli vedoucího projektu. Má práce na pozici programátora spočívala zejména v back-end části, kde jsem implementoval byznys logiku, doménovou vrstvu a další nezbytné části.

The screenshot shows a web form titled "Create topic" with a blue header bar containing "Topics", "Category", and "Admin page". The form has several sections: "Choose category" with a dropdown menu showing "Java"; "Headline / Short description" with a text input field containing "Some short description"; "Description" with a rich text editor containing placeholder text and some bolded text; "Resource" with a file upload button and a "Submit" button at the bottom right. The footer contains "Report bugs or comment our application" and "©2015 Tieto. All rights reserved. (version 1.0)".

Obrázek 7: Tieto Component shop - Vytváření nového příspěvku

4 Komerční projekt

Komerčním projektem se rozumí projekt, ze kterého má společnost určitý finanční zisk. V tomto projektu jsem zůžilkoval získané vědomosti z nekomerčních. V komerční části již není místo pro velké množství seberealizace z pohledu implementace. Veškeré aktivity, které jsem prováděl, byly funkční požadavky reálných klientů.

4.1 Tieto TRIP

K tomuto projektu jsem nastoupil okamžitě po vypracování posledního nekomerčního projektu *Tieto Component shop*, kdy během jeho implementace jsem absolvoval s dalším studentem interní pohovor právě do týmu pro Tieto TRIP.

Tieto TRIP je unikátní produkt kombinující prvky NoSQL databáze a vyhledávacího stroje. Byl vyvinut před více než dvěma desítkami let, kdy o nestrukturovaných datech a vyhledávání v nich mluvila jen akademická sféra [8]. Tieto TRIP poskytuje možnost ukládání v textových i multimediálních (video, obrázky) formátech [9].

Řešení založené na Tieto TRIP je nemalé množství, ale mezi nejběžnějšími jsou:

- deníky a žurnálovací systémy
- noviny a mediální archívy
- knihovní systémy
- patentové systémy
- vizuální archívy
- a další různé multimediální úložiště

V dnešní době Tieto TRIP používá více než padesát zákazníků a OEM partnerů ze zemí jako je Finsko, Švédsko, Norsko, Čína či USA [8]. Tito zákazníci pracují v různých odvětvích od zdravotnických organizací, veřejného sektoru, bankovního a finančního sektoru, až po telekomunikační, mediální a transportní společnosti. Na oficiálních stránkách produktu jsou uvedeni tito zákazníci [9]:

- NSTL - poskytuje dva Terabajty vědeckých a technických bibliografických záznamů průmyslu, vysokým školám, výzkumným ústavům a výzkumným pracovníkům.
- Alma Media - vyžaduje nejrychlejší přístup k referencím během psaní článků.
- Tronder Avis - poskytuje archív dvanácti miliónů článků a tří set tisíc obrázků v pěti stech tisících novinových člancích pro každého, kdo má zájem
- Finská imigrační služba - autorizovaní uživatelé vyhledávají informace v rámci třiceti tisíc dokumentů v textovém nebo multimediálním formátu.

4.1.1 Webová aplikace pro získání zkušební licence

Mým úvodním úkolem v novém týmu byla webová aplikace pro získání zkušební licence. Tato licence slouží pro prozkoumání Tieto TRIPu potencionálními zákazníky, studenty a jinými lidmi. Díky této časově omezené licenci je uživatel schopen pracovat po dobu třiceti dní se systémem tak, aby si vyzkoušel různé funkcionality popsané na oficiálních stránkách produktu. Na tomto úkolu jsem pracoval společně s dalším studentem, který byl přijat spolu se mnou a prostřednictvím vývoje jsme se měli seznámit s prostředím a celkově s prací s API od Tieto TRIP.

Tato aplikace byla napsána pomocí frameworku Spring jako Spring MVC, se kterým jsem neměl doposud žádné zkušenosti. Další novinkou pro mě byla NoSQL databáze, se kterou se pracuje nestandardně oproti běžným SQL databázím. Jedním z požadavků bylo využít právě Spring MVC, kdy jsem se měl naučit, jak se provádí vývoj, strukturování projektu a logicky odděluje datový model od uživatelského rozhraní, a jak se logicky řídí. Další z požadavků bylo využití interního API Tieto TRIPu napsaného v Javě, protože veškeré mé budoucí úkoly souvisí s jeho znalostí. Zajímavou částí implementace bylo využití antispamového mechanismu reCAPTCHA od společnosti Google. Tento snadno použitelný mechanismus zajišťuje vytvoření dvou klíčů (serverový a klientský) pro uživatelský Google účet. Další požadavek byl v jednoduchosti napojení do existujícího Wiki systému Confluence od firmy Atlassian.

Nejdůležitější použité technologie: Java 1.8, Spring framework ve verzi 4.2.1.RELEASE, Atlassian Confluence a další

Shrnutí a časové vyjádření Původně měla implementace trvat okolo patnácti dní, avšak dokončili jsem ji za čtyři dny, protože původní plán počítal s tím, že se nebudeme schopni tak rychle adaptovat. Opak byl pravdou a my jsme předčili veškeré odhady zadavatele. Během těchto dnů jsem se tedy seznámil se základy Tieto TRIPu, rozšířil své vědomosti Spring frameworku o Spring MVC, vyzkoušel si implementaci reCAPTCHA a seznámil se s Atlassian Confluence.

4.1.2 Transformace dat

Dalším méně rozsáhlým, avšak neméně důležitým úkolem, který jsme spolu s kolegou studentem dostali, byla transformace dat pro potřeby vyhledávání nad velkým množstvím dat. Velkým množstvím dat se zde rozumí několik desítek Gigabajtů dat. Tato data jsme získali prostřednictvím internetu z veřejně dostupných medicínských repositářů. Jednalo se o různé medicínské články o nemocech, léčích a lékařství celkově.

Hlavní požadavek bylo efektivně transformovat data ze stažených souborů do souborů vhodných pro databázový import. Stažená data byla ve formátu NXML, a proto byla nutná XSLT transformace do specifického formátu TFO, který používá Tieto TRIP. Pro tuto transformaci jsme využili Xalan procesor, kdy jsme museli vytvořit XSLT soubor, definující předlohu pro transformaci XML vstupního souboru. Výstupem této transformace byl tedy TFO soubor, který

bylo možné importovat do databáze. Následně bylo nutné vytvořit samotný kód v Javě, na kterém jsem se přímo nepodílel, ale kolega se mnou jeho řešení konzultoval. Následně bylo kód nutné optimalizovat, jelikož jeho běh byl zdlouhavý.

Nejdůležitější použité technologie: Java 1.8, Xalan procesor, XSLT pro transformaci XML a další

Shrnutí a časové vyjádření Pro tuto aplikaci nebyl určen žádný časový limit, jelikož nebylo možné přesně určit, za jak dlouhou dobu a jakým způsobem bude transformace probíhat. Já jsem na tomto úkolu strávil přibližně šest pracovních dní, kdy jsem se seznamoval s XSLT transformací pomocí Xalan procesoru a se strukturou TFO souboru, který akceptuje Tieto TRIP databáze pro import. Jakmile zbývalo dopsat samotný kód pro transformaci, tak jsem byl přeřazen na další úkol, a to na *Lokalizace GUI*. Po skončení implementace jsem na žádost kolegy provedl revizi kódu a refactoring, který zkrátil potřebný čas k transformaci o několik desítek procent.

4.1.3 Lokalizace GUI

Tento úkol mi byl přidělen během implementace transformace dat. Jednalo se o kompletní lokalizaci (překlad) grafického uživatelského rozhraní ve formě webu (více v kapitole *GUI pro Tieto TRIP*), kdy vznikl požadavek pro možnost změny různých národních jazyků.

Pro lokalizaci jsem využil vestavěnou funkcionalitu Spring frameworku, kde bylo nutné vytvořit soubor s názvem `messages_{požadovaná lokalizace (např. EN)}.properties`. Dále vytvořit lokalizační resolver (Cookies nebo Session) a interceptor (LocaleChangeInterceptor) pro umožnění změny jazyku. Lokalizace byla rozdělena na lokalizaci JSP stránek, JavaScriptu a Java kódu. Lokalizace **JSP stránek** byla implementována pomocí speciálního balíčku ze Spring frameworku, prostřednictvím kterého je možné získat tzv. zprávu, která reprezentuje řetězec z properties souboru. Dále lokalizace **JavaScriptu** pomocí javascriptové proměnné, která reprezentuje stejný stringový řetězec jako v JSP stránkách. Když bylo potřeba někde použít tento javascriptový soubor, muselo se nejprve deklarovat několik proměnných pro překlad. Lokalizace **Java kódu** byla vyhotovena využitím nově vytvořené servisní třídy, která pracovala se třídou `ResourceBundle`.

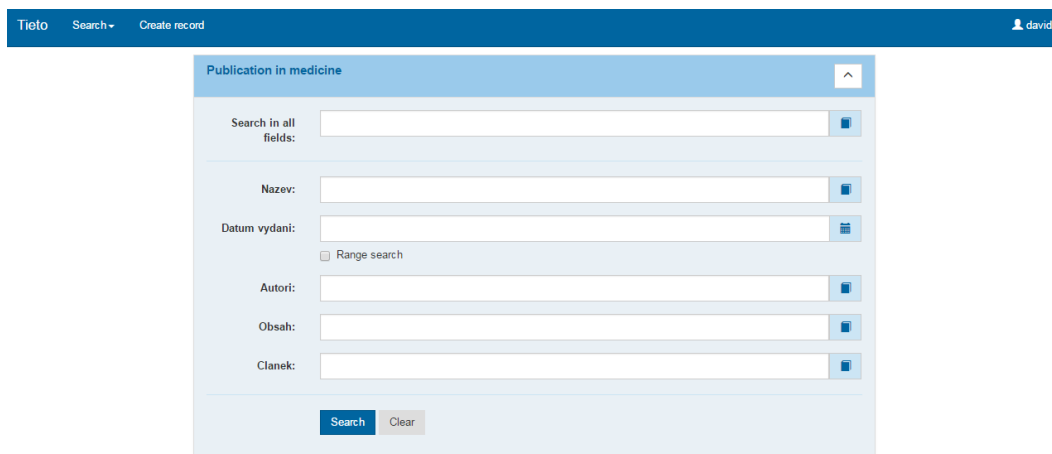
Nejdůležitější použité technologie: Java 1.8, Spring framework a další

Shrnutí a časové vyjádření Mým úkolem bylo během desíti pracovních dnů provést lokalizaci veškerých staticky definovaných textových řetězců ve webové aplikaci na dynamické textové řetězce z properties souborů. Tento čas byl dostačující a během implementace jsem se naučil pracovat s properties soubory, lokalizačními komponentami Spring frameworku a `ResourceBundle` třídu pracující jako mapa s klíčem a hodnotou pro lokalizaci.

4.1.4 Konfigurační framework

Mým dalším rozsáhlejším úkolem pro Tieto TRIP byl konfigurační framework, který měl sloužit ke konfiguraci nového GUI. Hlavním požadavkem tohoto úkolu byla snadná rozšiřitelnost, kde bylo zapotřebí implementovat co nejobecnější algoritmus pro tuto funkcionalitu. Dalším požadavkem byla snadná pochopitelnost a nenáročnost použití. Veškeré tyto konfigurace mohly být použity pro databázi obecně, skupinu uživatelů nebo konkrétního uživatele a mělo být využito co nejvíce prostředků Tieto TRIP, který obsahuje vlastní autorizaci a autentizaci, vlastní správu paměti, vlastní správu rolí a další funkcionality.

Před začátkem implementace jsem dostal úkol navrhnout řešení a následně jej prezentovat senior programátorům. Pro snadnou čitelnost jsem navrhl zadávat konfiguraci v JSON formátu, který se ukládá jako text do databáze a dále položky jako databáze, skupinu, uživatele a typ konfigurace zadávat jako položky v záznamu databáze. Toto řešení bylo schváleno, a tak se mohlo přejít k implementaci.



Obrázek 8: Konfigurační framework - Formulář pro pokročilé vyhledávání

Implementace probíhala způsobem, kdy jsme vytvořili interface pro zastupování všech typů konfigurací a následně tento interface implementovali v každé konfiguraci samostatně. Následně byla vytvořena třída pro načítání této konfigurace z databáze, kde se musel zadávat typ konfigurace, což vedlo ke špatnému řešení v rozsáhlosti kódu. Bylo potřeba tedy změnit toto chování, a proto jsme využili generických typů, které veškeré zbytečné nebo redundantní chování změnili v jedno obecné. Výsledkem bylo zobecnění celého algoritmu a snadné napojení nových typů, které vznikly nad rámec těchto čtyř základních typů:

- Vytváření záznamů (CreateRecord)
- Výsledky vyhledávání (ResultList)
- Detail záznamu (RecordDetails)
- Formulář pro pokročilé vyhledávání (AdvancedSearch)

Vytváření záznamů (CreateRecord) Tato konfigurace symbolizuje předlohu pro vytváření nového záznamu do databáze. Uživatel, mající právo vytvářet konfigurace, definuje jednotlivé položky, kde každá položka reprezentuje sloupec v databázi. Pro tyto položky se může nastavit popis, který slouží jako vlastní název místo názvu z databáze, dále může definovat delší popis pro tooltip.

Výsledky vyhledávání (ResultList) Konfigurace nazývaná ResultList slouží pro definování hlavičky vyhledaných výsledků. Pokud je konfigurace správně nastavená a vyhledáváme například zdravotní údaje, zobrazí se nám vždy nejdříve ty položky, které byly nakonfigurovány, i když neobsahují žádná data. Navíc se nám zobrazí i položky, které nakonfigurovány nebyly, a to z důvodu výskytu vyhledávaného výrazu. ResultList má sloužit pro rychlou orientaci v databázi, nad kterou vyhledáváme.

Detail záznamu (RecordDetails) RecordDetails zastupuje detail konkrétního vyhledaného výsledku. Slouží výhradně k seřazení jednotlivých položek záznamu a pojmenovávání těchto položek stejně, jako tomu bylo u CreateRecord.

Formulář pokročilého vyhledávání (AdvancedSearch) Tato speciální konfigurace slouží pro vytvoření formuláře pro pokročilé vyhledávání (ukázka na obrázku 8: Konfigurační framework - Formulář pro pokročilé vyhledávání). Formulář může obsahovat položky, které často vyhledáváme, například u knih by to byly položky *autor*, *vydavatelství*, *název knihy* a implicitně vyhledávání ve všech polích. Tento formulář slouží pro přesné vyhledávání nad některými položkami databáze nebo clusteru (skupina databází). Pokročilý formulář ve formátu JSON definujeme například takto:

```
{  "fields": [
    { "type": "PhraseField", "referenceRestricted": false, "dbfieldName": "Title",
      "validValueList": null, "label": "Název", "description": "Název článku"},
    { "type": "DateField", "referenceRestricted": false, "dbfieldName": "DATE_day",
      "validValueList": null, "label": "Datum vydání", "description": "Vydáno"},
    { "type": "TextField", "referenceRestricted": false, "dbfieldName": "Abstract",
      "validValueList": null, "label": "Obsah", "description": "O tom to je"},
    { "type": "TextField", "referenceRestricted": false, "dbfieldName": "Body",
      "validValueList": null, "label": "Článek", "description": "článek samotný"}
  ],
  "formName": "Publication in medicine"
}
```

Výpis 1: Konfigurační framework - Formulář pro pokročilé vyhledávání

Jakmile byla dokončena implementace jádra konfiguračního frameworku, byl vznesen požadavek na propojení s GUI Tieto TRIP. Doposud veškeré činnosti, které měl vyřešit konfigurační framework byly zaneseny do aplikace fixně a nerozšiřitelně. Toto omezení jsme nahradili naším řešením a GUI aplikace se stala obecnější a méně závislá na použitelných komponentách. To jsme si ověřili při prvním rozšiřování konfigurace, kdy bez jakýchkoliv problémů bylo možné rozšířit doposud čtyři konfigurace o nové dvě, implicitní nastavení (DefaultConfig) a Thesaurus.

Implicitní nastavení (DefaultSettings) Tato konfigurace zastupuje implicitní nastavení aplikace, kde umožňuje nastavovat implicitní jazyk a vzhled, který může být obyčejný nebo korporátní. Ukázka tohoto nastavení je na obrázku 8: Konfigurační framework - Formulář pro pokročilé vyhledávání, kde lze spatřit korporátní vzhled a jeho JSON formát lze definovat takto:

```
{  
  "defaultLanguage": "en-EN",  
  "defaultTheme": "tieto"  
}
```

Výpis 2: Konfigurační framework - Implicitní nastavení pro GUI

Thesaurus Tato konfigurace slouží pro nastavení Thesauru v Tieto TRIP. Ten umožňuje použití Thesauru na databázi. V této konfiguraci musí uživatel definovat pouze databázi Thesauru (Thesaurus databáze je speciální databáze).

Po celkové implementaci nám byl přidělen další úkol, a to ukládání těchto konfigurací do mezipaměti (caching) pro aktuální relaci (session). Tento mechanismus zabráňuje zbytečným databázovým dotazům a snižuje tím režii databáze. Dále se samozřejmě objevili některé nesrovnalosti ve funkčních i nefunkčních attributech, takže bylo potřebné opravit tyto chyby. Následně byl tento úkol uzavřen jako hotový.

Nejdůležitější použité technologie: Java 1.8, Jackson JSON Processor a další

Shrnutí a časové vyjádření Dostali jsme úkol vytvořit konfigurační framework pro potřeby Tieto TRIP GUI aplikace. Tento framework měl načítat čtyři druhy konfigurací pro vytváření záznamů, výsledky vyhledávání, detail záznamu a formuláře pro pokročilé vyhledávání (později ještě implicitní nastavení a Thesaurus). Veškeré tyto konfigurace bylo možné nastavit pro jednotlivé databáze, skupiny uživatelů a uživatele samotné, kde data samotné konfigurace byly uloženy ve formě JSON. Postup této aplikace jsme museli s kolegou navrhnout a následně jej vyhotovit. Při implementaci jsme použili generické datové typy, takže jsem měl možnost si pořádně doslova "hrát" s těmito řešeními. Celkově nám projekt zabral okolo dvaceti pracovních dnů, do kterých počítám opravu chyb, rozšiřování funkcionality a zakomponování do GUI.

4.1.5 Další úkoly

Tato část si klade za cíl pouze zmínit některé méně časově náročné úkoly, do kterých začleňuji výhradně opravu chyb a refaktoring určitých částí kódu.

Jakmile se aplikace, na které se podílím, blížila k demonstraci u klienta, bylo potřebné opravit či upravit různé části kódu a funkcionality. Velkou část těchto úprav mi zabraly Javascriptové soubory, ve kterých se zasílají požadavky do jednotlivých controllerů, které následně tyto požadavky zasílají na server a vrací odpovědi. Právě Javascript měl vyřešit korektní interpretaci dosažených výsledků, ale taktéž reakci na chybové stavy serveru a aplikace jako celku. Tato funkcionality nebyla zcela správně vyřešena, a proto jsem dostal za úkol některé tyto stavy opravit. Další netriviální opravy byly potřebné v jádru Tieto TRIP nebo API, do kterých jsem neměl možnost zasahovat.

Dalším méně náročným úkolem, který trval kolem čtyř pracovních dní byla **interpretace serverové činnosti**, ukázka "něco se děje", jelikož celá aplikace je stavová (z důvodů implementace Tieto TRIP), je možné zaslat několik požadavků na server a server je musí všechny postupně vyřešit (absence více vláknových operací). Mým úkolem bylo ošetřit zaslání pouze jedné takovéto žádosti, při vyhledávání delším než tři sekundy a zobrazení tzv. spinneru a časového údaje.

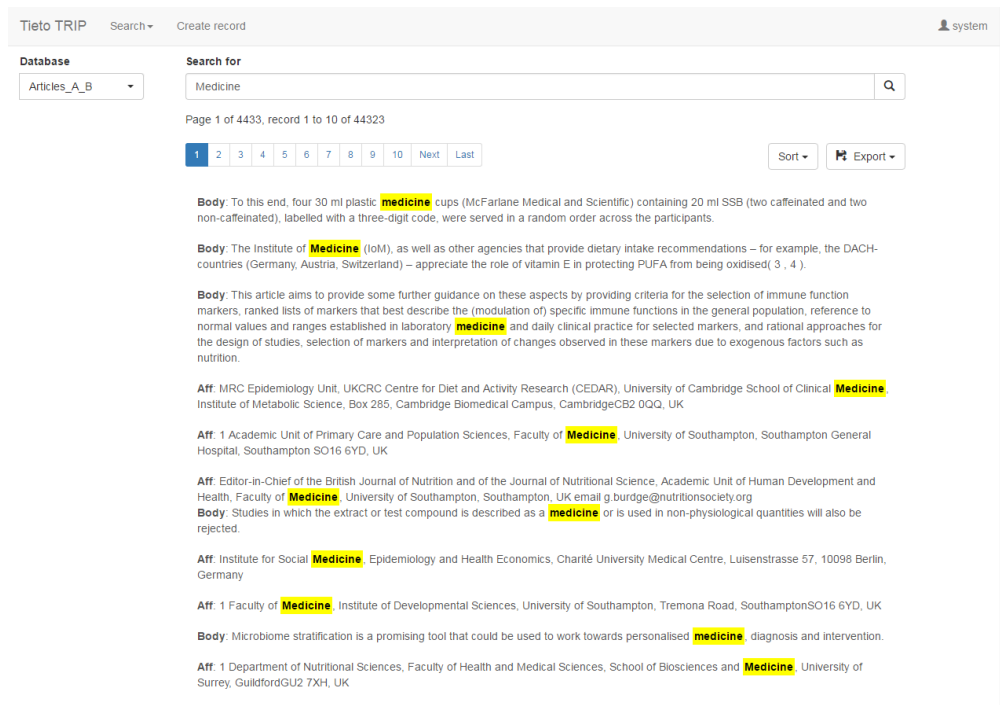
Refaktoring a optimalizace kódu, které jsem prováděl byly nedílnou součástí různých implementací, a proto je chci zmínit jako část úkolů mé práce pro Tieto TRIP. Díky těmto dvěma skutečnostem jsem měl možnost procházet cizí kód a pokusit se jej optimalizovat. Ve většině těchto případů se mi to povedlo s následným zrychlením provádění úkonů, pro které byly tyto algoritmy vymyšleny.

Dalším z řady úkolů byla implementace **třídění vyhledaných výsledků**, kde byl požadavek třídění výsledků vyhledávání. Tuto funkcionality jsem musel s kolegou prozkoumat a zjistit jak interně funguje. Následně jsem implementoval různé Javascriptové akce spolu s novou implementací v Java kódu. Taktéž bylo nutné přesně provést úpravu stylu tohoto řešení, aby bylo co nejvíce intuitivní a snadné k použití.

Shrnutí a časové vyjádření Do této sekce zapadají veškeré méně náročné a nerozsáhlé řešení, mezi které patří například refaktoring, oprava chyb, třídění záznamů, úprava stylu aplikace, implementace Javascriptových souborů, optimalizace kódu apod. Celkový čas strávený na této části projektu se nedá lehce shrnout, ale pohybuje se v rozmezí dvaceti až třiceti pracovních dní. Během této práce jsem se seznámil s velkým množstvím detailů týkajících se Javascriptu, Spring frameworku, architektury aplikací apod., čímž jsem prohloubil své znalosti.

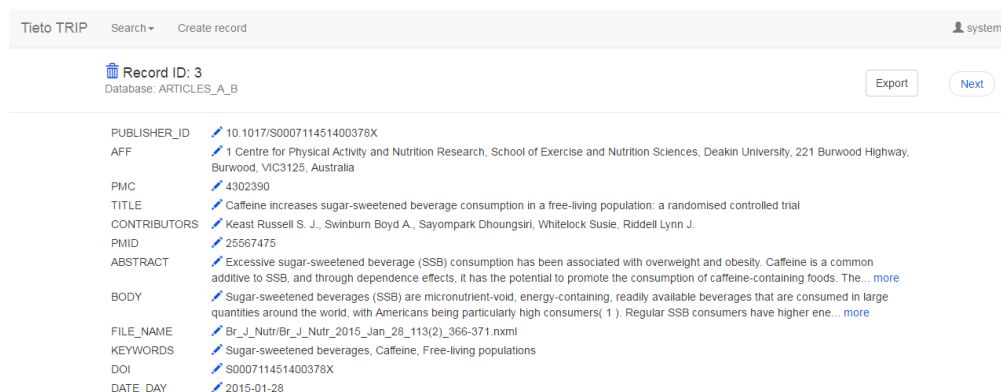
4.1.6 GUI pro Tieto TRIP

Tieto TRIP byl doposud implementován jako konzolová aplikace (klasický příkazový řádek nebo MMC). Kvůli měnícím se trendům vznikl požadavek na vytvoření nového GUI, které by mělo nahradit dosavadní. Nové grafické uživatelské rozhraní je postaveno na bázi webové aplikace, která umožňuje všestranné použití, z důvodů nutnosti instalace pouze webového prohlížeče, snadnou rozšiřitelnost a modernizaci Tieto TRIP. Ukázku moderního GUI lze vidět na obrázku 9: Tieto TRIP GUI - Základní vyhledávání slova.



Obrázek 9: Tieto TRIP GUI - Základní vyhledávání slova

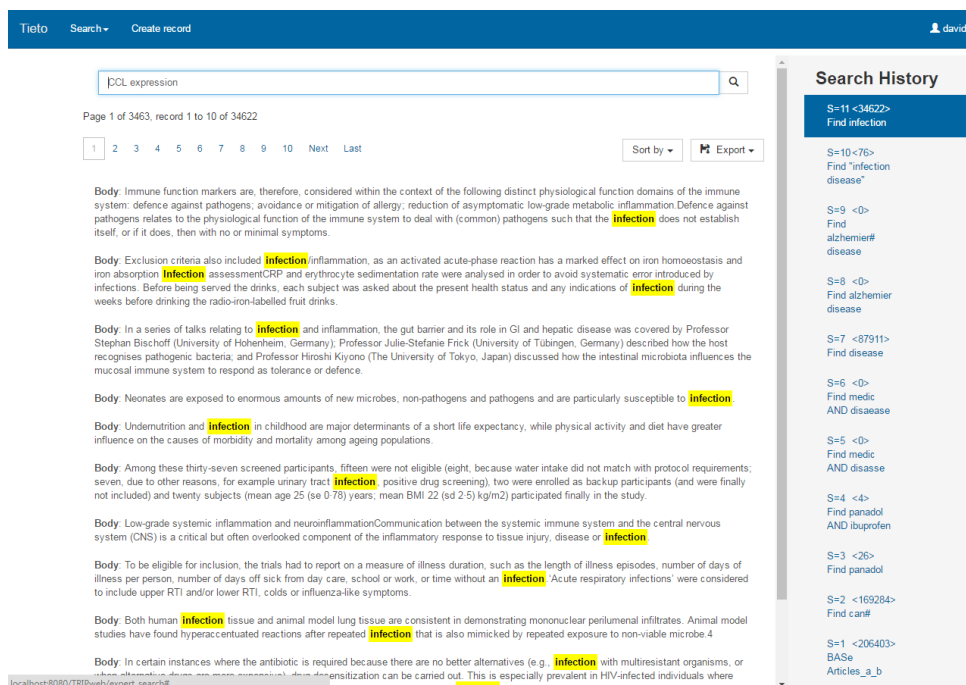
Vývoj nového GUI je prozatím v počátcích, kdy byla implementována pouze nezbytná funkcionality, která zahrnuje vyhledávání v několika módech. Základní mód umožňuje rychle vyhledávání s menší specifikací vyhledávaných informací. Pokročilý mód umožňuje vyhledávat přímo nad některými položkami databáze (například pro knižní databázi: kapitola a autor) a expertní mód, který je možné vidět na obrázku 11: Tieto TRIP GUI - Expertní mód, vyžaduje znalost příkazů Tieto TRIPu. Prostřednictvím expertního vyhledávání je uživatel schopen provádět veškeré příkazy produktu. Dále GUI zjednodušuje vytváření záznamů, editaci, export, detailní zobrazení výsledku (ukázka na obrázku 10: Tieto TRIP GUI - Detail vyhledané položky slova) a další pokročilejší úkony plynoucí z použití právě webové aplikace. Nové prvky v systému je nutné vytvářet a upravovat v samotném jádře produktu, které je napsáno v jazyce C a C++. Z důvodu komplexnosti a rozsáhlosti softwaru, je nesnadné postupovat rychle v tomto rozšiřování.



Obrázek 10: Tieto TRIP GUI - Detail vyhledané položky slova

Nejdůležitější použité technologie: Java 1.8, Spring framework, Tieto TRIP, Bootstrap, jQuery a další

Shrnutí a časové vyjádření Na této aplikaci jsem pracoval nepřetržitě od přidělení do týmu Tieto TRIP. Mezi mou práci patřila implementace nové funkcionality (*Lokalizace GUI*, *Konfigurační framework* apod.), oprava chyb (*Další úkoly*) a příprava data pro import (*Transformace dat*). Velice důležité bylo seznámení se s Tieto TRIP, jelikož je to komplexní software s rozsáhlou funkcionalitou. Nové trendy vývoje GUI přiměly vedoucího produktu zmodernizovat také GUI pro Tieto TRIP a tím znovu proniknout do povědomí klientů.



Obrázek 11: Tieto TRIP GUI - Expertní mód

5 Závěr

5.1 Uplatněné a scházející dovednosti

Během celého průběhu praxe jsem pracoval s platformou Java na vývoji informačního systému, tudíž jsem nejvíce využíval znalostí z předmětů *Programovací jazyky I* a *Vývoj informačních systémů*. Úroveň, kterou jsem získal předmětem *Programovací jazyky I*, nebyla dostačující, a proto jsem se musel hlouběji seznámit s několika věcmi, mezi které bych zařadil generické datové typy, moderní prvky a práci s již existujícími třídami Javy verze 1.8. Předmět *Vývoj informačních systémů*, který mě měl naučit architekturu informačních systémů, jsem využíval neustále. Téměř v každém projektu jsme pracovali s různými návrhovými vzory. Základem obou těchto předmětů byl předmět *Programování II*, kdy jsem veškeré znalosti nabyté tímto předmětem využil. Další vědomosti jsem uplatnil z předmětu *Algoritmy II*. K práci s databázemi jsem využil znalosti z předmětů *Úvod do databázových systémů* a *Databázové a informační systémy*. Další významné předměty: *Úvod do softwarového inženýrství*, *Uživatelské rozhraní*, *Kompetence pro trh práce*

Mezi scházející dovednosti bych určitě zařadil absenci předmětů, kde se pracuje v týmech, jelikož téměř vždy má programátor možnost konzultovat řešení úkolu s kolegy. Další klíčovou scházející znalostí byla práce s verzovacími systémy, ale tato neznalost souvisí s prací v týmu. Během mé praxe se plně projevila absence lepší úrovně anglického jazyka.

5.2 Zhodnocení odborné praxe a jejích výsledků

Tato praxe byla a je jednou z nejdůležitější částí mého studia na vysoké škole. Splnila přesně to, co jsem od ní očekával a možná ještě něco více. Hlavním přínosem bylo propojení nabytých teoretických zkušeností, kdy jsem měl možnost vyzkoušet si nespočet technologií, přístupů, a také vyřešit problémy, se kterými se v akademické sféře nesetkám.

Během praxe jsem se podílel na nekomerčních projektech, které mě měly naučit co nejvíce nových technologií a přístupů. Taktéž mě naučily pracovat v týmu a přejímat zodpovědnost za kvalitu mých řešení. Seznámil jsem se s některými agilními technikami vývoje softwaru a vyzkoušel různé přístupy k programování.

Po těchto nekomerčních projektech jsem přešel na komerční projekt Tieto TRIP. V tomto týmu zkušených programátorů jsem se naučil mnoho zajímavých věcí. Ze začátku byly mé a kolegovy schopnosti pro tým neznámé, a proto jsme úkol *Webová aplikace pro získání zkušební licence* dokončili rychleji, než bylo požadováno. Následně jsme dostávali těžší úkoly a dostali jsme příležitost podílet se na implementaci klientských požadavků nové aplikace.

Společnost si prostřednictvím této praxe mohla zaškolit studenta "na míru", což vedlo ke snížení zatížení senior programátorů, kteří se tak mohli soustředit na složitější úkoly. Spolu s dalším studentem jsme tak jediní junior programátoři v tomto týmu. Po skončení praxe budu pravděpodobně na této pozici nadále působit, jelikož jsem prokázal své technické schopnosti.

Literatura

- [1] Historie společnosti Tieto. Tieto. [online]. 2015 [cit. 2016-03-01]. Dostupné z:
<http://www.tieto.cz/tieto-o-nas/historie-tieto-czech-republic>
- [2] Part I. Overview of Spring Framework. Spring Framework Reference Documentation. [online]. 2004-2015 [cit. 2016-03-05]. Dostupné z:
<https://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#spring-introduction>
- [3] Spring Tool SuiteTM. Spring. [online]. 2016 [cit. 2016-03-05]. Dostupné z:
<https://spring.io/tools>
- [4] Getting Started. Primefaces. [online]. 2009-2014 [cit. 2016-03-06]. Dostupné z:
<http://www.primefaces.org/gettingStarted>
- [5] Welcome to Apache Maven. Apache Maven Project. [online]. 2002–2016 [cit. 2016-03-15]. Dostupné z: <https://maven.apache.org/>
- [6] Spring Roo. Spring. [online]. 2016 [cit. 2016-03-16]. Dostupné z:
<http://projects.spring.io/spring-roo/>
- [7] JIRA Software. Atlassian. [online]. 2016 [cit. 2016-03-16]. Dostupné z:
<https://www.atlassian.com/software/jira>
- [8] Tieto TRIP. Tieto – Kariéra. [online]. 2016 [cit. 2016-03-20]. Dostupné z:
<https://jobs.tieto.cz/projects/tietotrip/>
- [9] What is Tieto TRIP. Tieto TRIP. [online]. 2015 [cit. 2016-03-20]. Dostupné z:
<https://trip.service.tieto.com/wiki/display/WITT/What+is+Tieto+TRIP>